# Introduction

Java graphics classes, ***SciGraph***, were developed to allow a NOAAServer user to interactively preview and overlay plots of more than one dataset at a time, including datasets that are stored on **different** Local Data Servers.

# Design Goals

- Allow a graphics client developer a great deal of flexibility and freedom.
- GIS style layer approach to display geophysical data.
- Support several types of graphical display.
  - X-Y plot.
  - 2-D contour and “pixel” plots.
  - Vector plots.
  - Point-Value plot.
- Develop a framework that is easily extended.
- Not a general purpose graphics package, but a set of tools.

# Mouse Events

**Two basic types of mouse events are supported by SciGraph.**

- The mouse can be used to select SciGraph objects. The developer determines how the application then interacts with the object.
- The mouse can also be used to specify a rectangle on the **Pane**. Again, the developer determines how this information is used by the application/applet.

# Overview

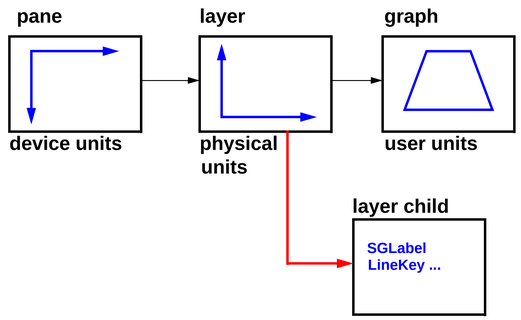
## SciGraph has three main components.

- The **Pane**, on which all graphics are drawn.
- The **Layer**, which insulates the developer from the “device” coordinates of the Pane to “physical” coordinates.
- The **Graph**, which provides the transforms from “user” coordinates to “physical” coordinates.

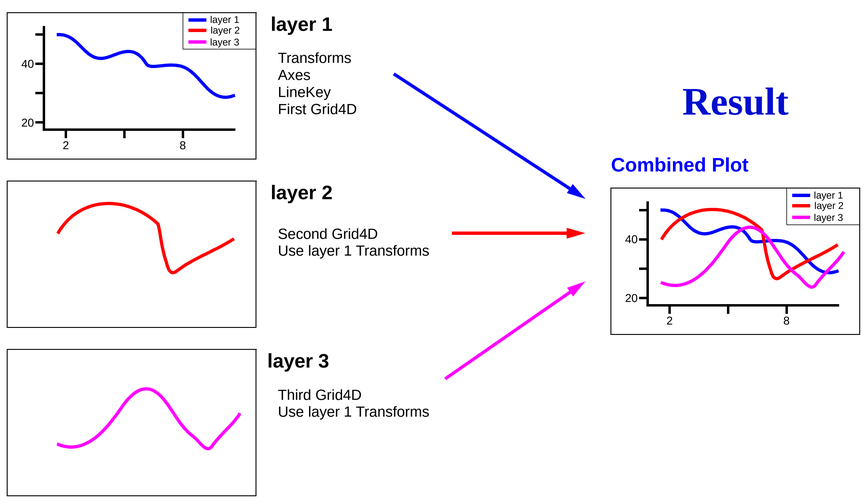
## Additional components include:

- **Graph** classes. Line, vector, contour, etc...
- **Transforms** that specify the “user” to “physical” coordinates transformation.
- **Axes**. Linear, log, etc...
- **LayerChild** classes. Labels, keys, ...

# Main Graphical Components

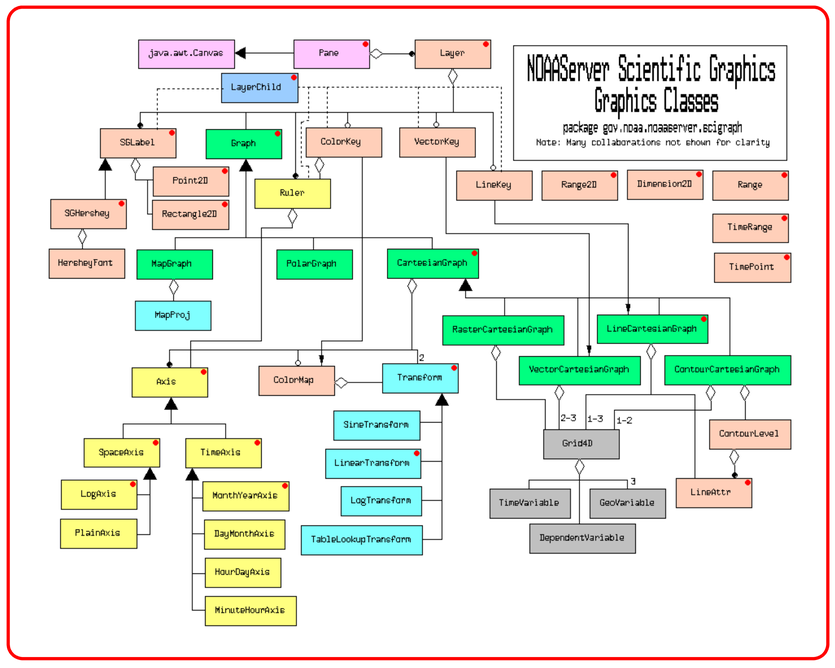



# An Example



# SciGraph: Object-Oriented 2D Scientific Graphics Library

Donald W. Denbo
*NOAA/Pacific Marine Environmental Laboratory*
*JISAO/Joint Institute for the Study of the Atmosphere and Ocean*




## Principle Classes

- **Pane**. Derived from java.awt.Canvas.
- **Layer**. Transforms “physical” to “device” coordinates.
- **Graph**. Abstract base class for all graphics.
  - **CartesianGraph**. Supports transforms where y’ = f(y), x’ = g(x).
    - **LineCartesianGraph**. Line plot style is supported.
    - **ContourCartesianGraph**. Contour plot style is supported.
    - **VectorCartesianGraph**. Vector plot style is supported.
    - **RasterCartesianGraph**. Raster plot style is supported.
  - **MapGraph**. Supports map transforms, e.g. polar stereographic, lambert, etc.
  - **PolarGraph**. Supports the polar transform.

## LayerChild Classes

**The LayerChild interface defines how graphical objects are associated with a Layer.**

- **SGLabel**. Draws text on a layer object.
- **LineKey**. Associates the appearance of a line with a label.
- **VectorKey**. Relates the size of a vector to its magnitude.
- **ColorKey**. Relates a ColorMap to user values graphically.

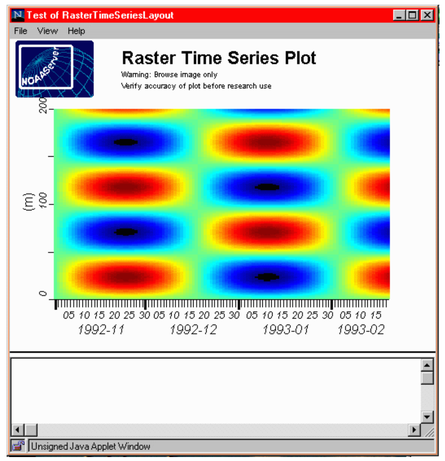
## Classes used with CartesianGraph

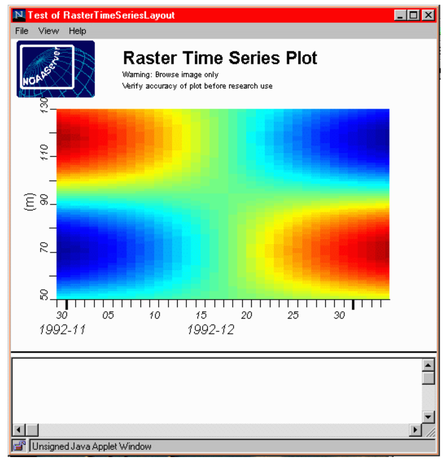
- **Axis**. Base class for cartesian axes.
  - **SpaceAxis**. Base class for float valued axes.
    - PlainAxis.
    - LogAxis.
  - **TimeAxis**. Base class for time axes.
    - MonthYearAxis
    - DayMonthAxis
    - HourDayAxis
    - MinuteHourAxis
- **Transform**. Base class for cartesian transforms.
  - LinearTransform.
  - SineTransform.
  - LogTransform.
  - TableLookupTransform.

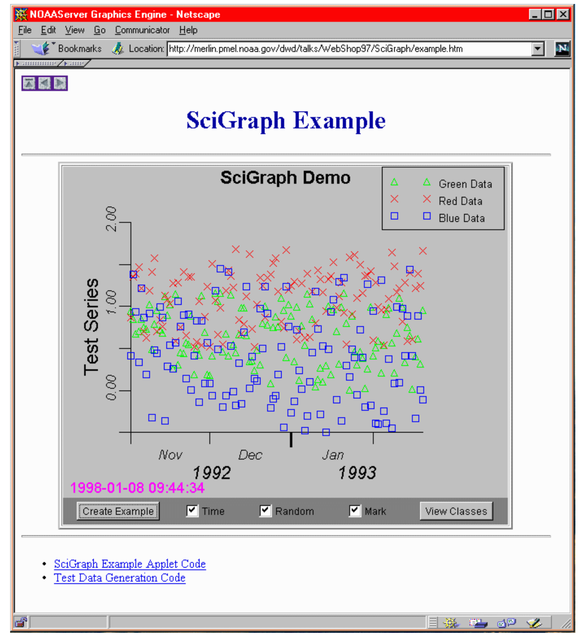
## Some Utility Classes

- **LineAttr**. Defines line style for LineCartesianGraph and ContourCartesian-Graph.
- **TimeRange**. Contains minimum and maximum times.
- **Range2D**. Contains minimum, maximum, and delta float values.
- **ColorMap**. Defines a lookup colormap.

*Zoom*



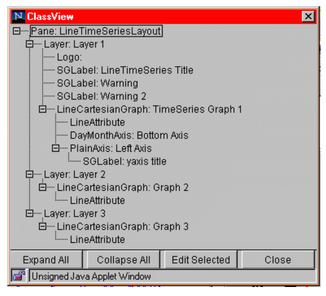



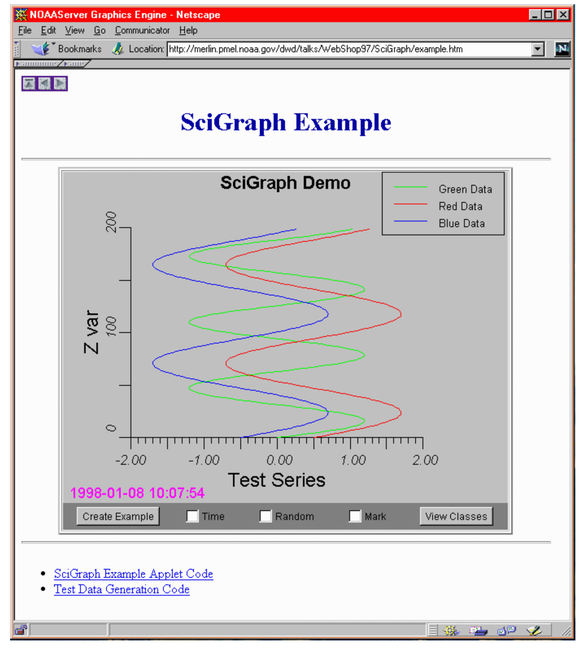







*SciGraph Class Tree*




## Summary

- SciGraph performance is good on Java VM's with Just-In-Time compilers.
- SciGraph has been successfully used in the development of the NOAAServer Co-plotting Prototype.
- SciGraph has proven to be very flexible.

## What's Next?

- Add more mouse event handling. For example, data point selection using the mouse.
- Complete implementation of the CartesianGraph subclasses. This includes graph classes, axes, transforms, and keys.
- Design MapGraph class and its subclasses.
- Modify NOAAServer graphics classes to fully implement the **JavaBean** standard.